Plug-and-Play Distributed Synthesis and Computation of Predictive Controllers

Colin N Jones

Ye Pu

Christian Conte

Melanie Zeilinger

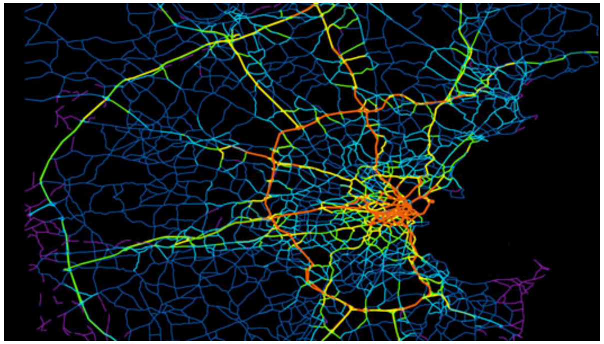
Laboratoire d'Automatique

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

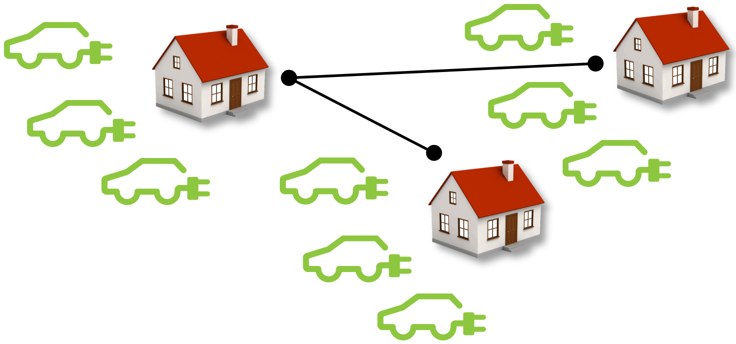
Large-Scale Distributed Systems

Motivation: Control large system composed of interconnecting sub-systems

Traffic networks



Cooperative robots



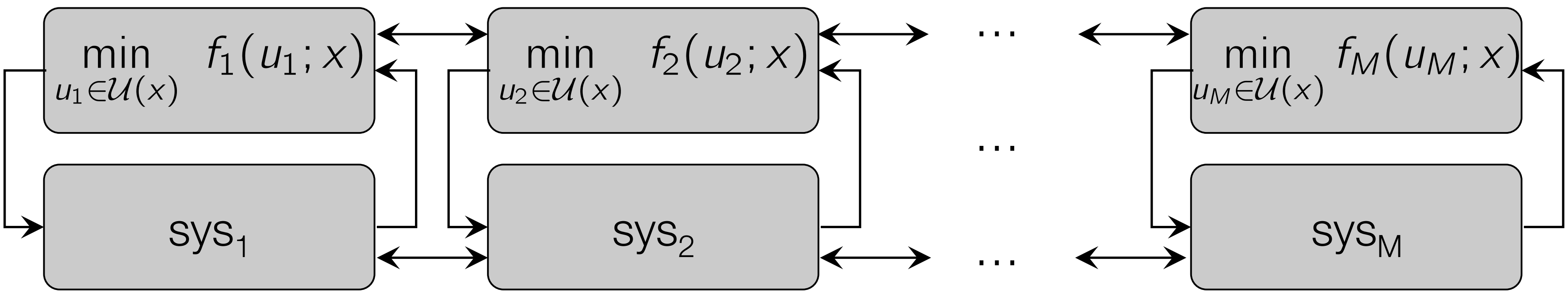
Building networks / Smart grids

- Systems: Buildings / Cars / Solar panels
- Coupling: Thermal / Electrical / Economic

Distributed Model Predictive Control (MPC)

Cooperative objective
$l(x, u) = \sum_{i=1}^{M} l_i(x_i, u_i)$

Communication with neighbours



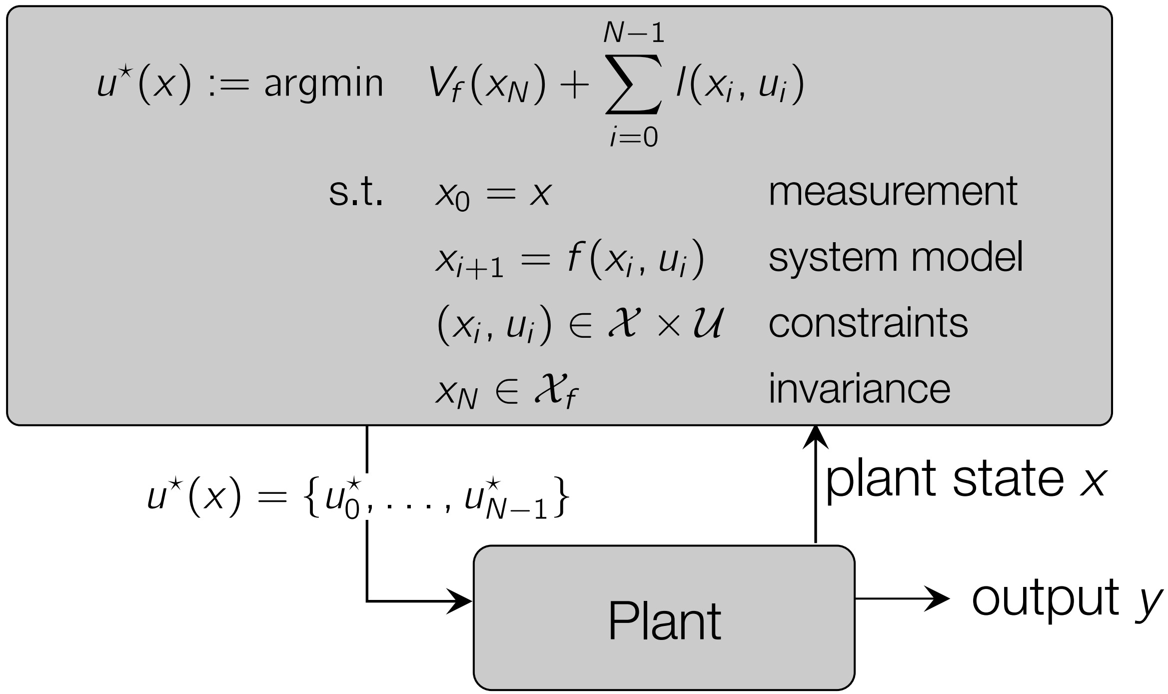
Independent constraints
$x_i \in X_i \quad u_i \in U_i$

Coupled linear dynamics
$x_i^+ = \sum_{j=1}^{M} A_{ij} x_j + B_i u_i = A_{\mathcal{N}_i} x_{\mathcal{N}_i} + B_i u_i$

This talk: Towards two key questions

Design:	Ensure stability and constraint satisfaction
Computation:	Fully distributed design, synthesis and control

Model Predictive Control (MPC)



MPC theory:

- ☺ Recursive constraint satisfaction
- ☺ Stability by design

MPC computation:

- ☺ Ultra-fast convex solvers
- ☺ ~Hard real-time implementations

Distributed Model Predictive Control (MPC)

Centralized MPC theory:

☺ Recursive constraint satisfaction

☺ Stability by design

Established approach:

- Terminal invariant set
- Terminal Lyapunov cost

This talk:

- Non-trivial terminal conditions
  - Distributed dynamic invariant sets
  - Distributed synthesis
  - Plug-and-Play

☺ Larger region of attraction

Distributed MPC:

Difficult to apply terminal conditions

1. No terminal conditions
   (Unknown region of attraction)
2. Trivial terminal condition
   (Very small region of attraction)

Outline

Part I

Distributed dynamic invariant sets

☺ Larger region of attraction

Part II

Plug-and-play

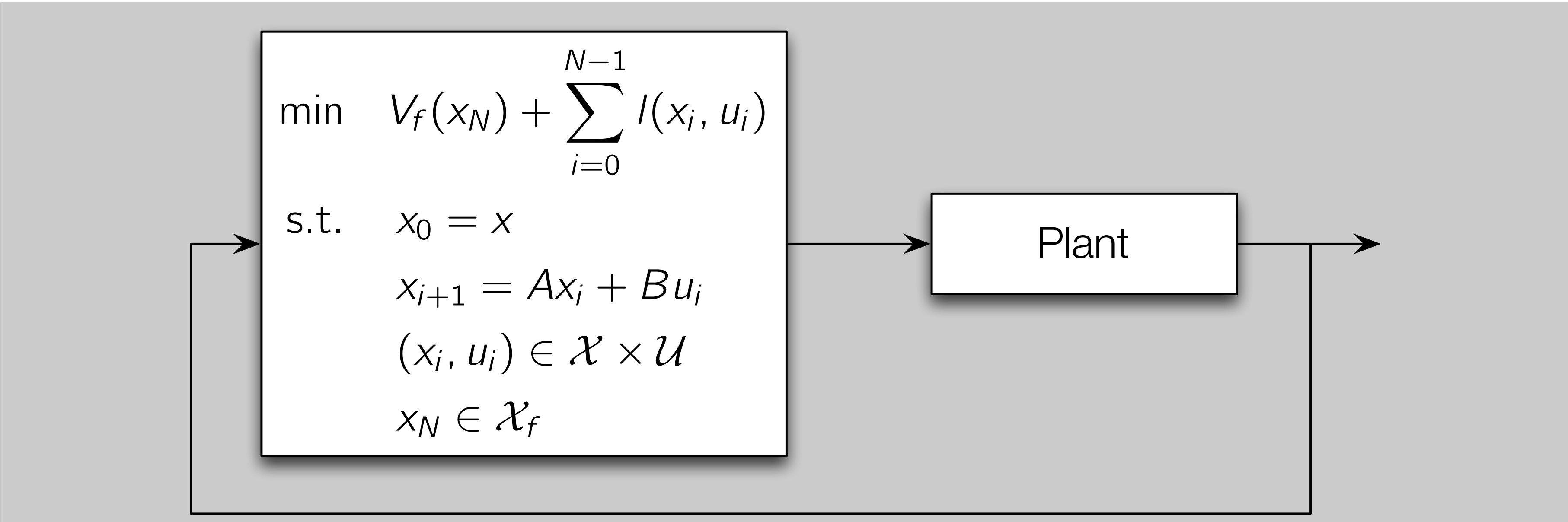
☺ Adapt to changing networks

Part III

Accelerated distributed optimization

☺ (Towards) Real-time MPC

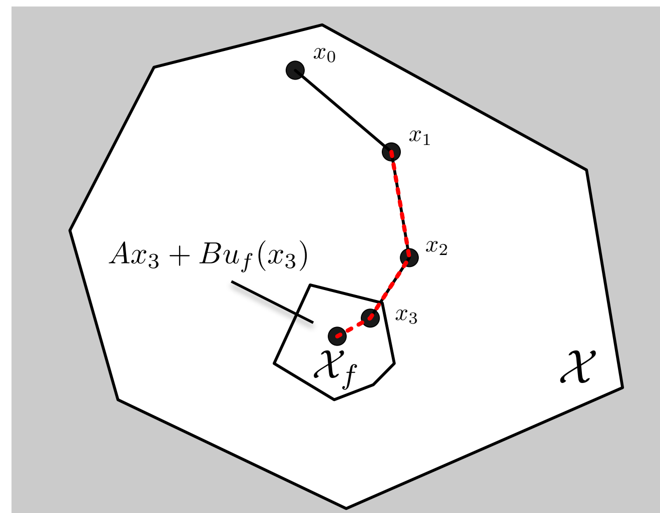
Stability and Invariance of MPC



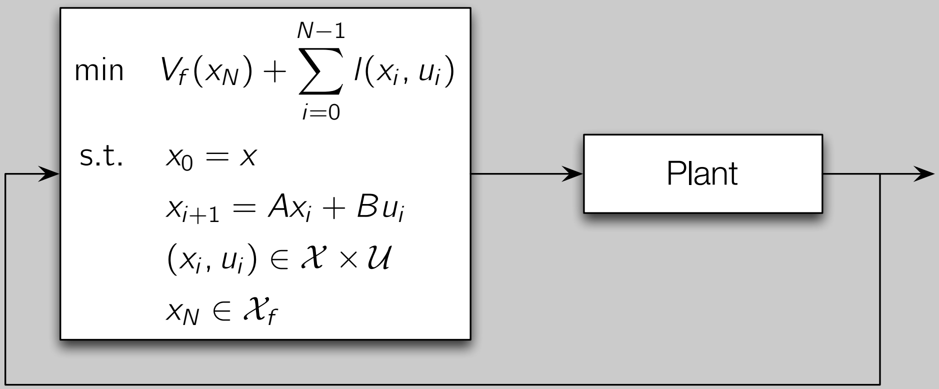
Stability and invariance if:

1. $\mathcal{X}_f \subset \mathcal{X}$ is invariant
   $x \in \mathcal{X}_f \Rightarrow Ax + Bu_f(x) \in \mathcal{X}_f$

2. $V_f(x)$ is a Lyapunov function in $\mathcal{X}_f$
   $V_f(Ax + Bu_f(x)) - V_f(x) \leq -l(x, u_f(x))$



Two Conflicting Requirements



$A, \; B$ structured

$\mathcal{X}, \; \mathcal{U}$ distributed

$l(x, u)$ distributed

1 Stability and invariance if:

1. $\mathcal{X}_f \subset \mathcal{X}$ is invariant — **Dense**
   $x \in \mathcal{X}_f \Rightarrow Ax + Bu_f(x) \in \mathcal{X}_f$
2. $V_f(x)$ is a Lyapunov function in $\mathcal{X}_f$ — **Dense**
   $V_f(Ax + Bu_f(x)) - V_f(x) \leq -l(x, u_f(x))$

2 Structured optimization

Terminal cost & constraints:

$$\mathcal{X}_f = \mathcal{X}_f^1 \times \cdots \times \mathcal{X}_f^M$$

$$V_f(x) = \sum_{k=1}^{M} V_f^k(x_{\mathcal{N}_k})$$

$$u_f(x) = [u_f^1(x_{\mathcal{N}_1}), \ldots, u_f^M(x_{\mathcal{N}_M})]^T$$

Goal: Satisfy both requirements

Requirement: No central coordination
(Online & offline optimization to have same coupling structure as system)

Structured Lyapunov Function

Lyapunov requirement: $V_f(x^+) < V_f(x)$

Structure requirement: $V_f(x) = V_f^1(x_1) + \cdots + V_f^M(x_M)$

Local Lyapunov decrease in each subsystem sufficient for stability:

$$V_f^i(x_i^+) < V_f^i(x_i), \quad \forall i \in \{1, \ldots, M\}$$

$$V_f^i\left(\overbrace{A_{ii}x_i + B_i u_f^i(x_{\mathcal{N}_i}) + \sum_{j \in \mathcal{N}_i} A_{ij}x_j}\right) < V_f^i(x_i)$$

Very conservative; often impossible in presence of strong dynamic coupling

Idea: Allow local increase while requiring a global decrease

Structured Lyapunov Function

Lyapunov requirement: $V_f(x^+) < V_f(x)$

Structure requirement: $V_f(x) = V_f^1(x_1) + \cdots + V_f^M(x_M)$

Idea: Allow local increase while requiring a global decrease

$V_f(x) := \sum_{i=1}^{M} V_f^i(x_{\mathcal{N}_i})$ is a Lyapunov function if

$$V_f^i(x_i^+) < V_f^i(x_i) + \gamma_i(x_{\mathcal{N}_i})$$

$$\sum_{i=1}^{M} \gamma_i(x_{\mathcal{N}_i}) \leq 0$$

[Jokic, Lazar, 2009]

Possible local increase

Global decrease

Dynamic Invariant Set

Level sets of a Lyapunov function are invariant:

$$\mathcal{X}_f = \left\{ x \;\middle|\; V_f(x) = \sum_{i=0}^{M} V_f^i(x_{\mathcal{N}_i}) \leq \hat{\alpha} \right\}$$

Problem: This terminal constraint couples all sub-systems

Want a condition that can be tested in a distributed fashion

$$\mathcal{X}_f(\bar{\alpha}) = \mathcal{X}_f^1(\alpha_1) \times \cdots \times \mathcal{X}_f^M(\alpha_M)$$

$$\mathcal{X}_f^i(\alpha_i) = \{x \,|\, V_f^i(x_{\mathcal{N}_i}) \leq \alpha_i\} \;\text{ where } \sum_{i=0}^{M} \alpha_i = \alpha$$

Problem: $\mathcal{X}_f^i(\alpha_i)$ is not invariant...

$$V_f^i(x_i^+) < V_f^i(x_i) + \gamma_i(x_{\mathcal{N}_i}) \not\leq V_f^i(x_i)$$

Dynamic Invariant Set

Define auxiliary dynamics, with the same structure as the system dynamics:

$$\alpha_i^+ = \alpha_i + \gamma_i(x_{\mathcal{N}_i})$$

Thm: Time-varying invariant set

$$x_i \in \mathcal{X}_f^i(\alpha) \Rightarrow x_i^+ \in \mathcal{X}_f^i(\alpha^+)$$

$$V_f^i(x_i^+) < V_f^i(x_i) + \gamma_i(x_{\mathcal{N}_i}) \leq \alpha_i + \gamma_i(x_{\mathcal{N}_i}) = \alpha_i^+$$

which implies global invariance and constraint satisfaction

If $\left\{x \mid \sum V_f^i(x_{\mathcal{N}_i}) \leq \sum \alpha_i \right\} \subseteq X$, then $\mathcal{X}_f(\bar{\alpha}) \subseteq X \Rightarrow \mathcal{X}_f(\bar{\alpha}^+) \subseteq X$

since $\sum \alpha_i^+ = \sum \alpha_i + \sum \gamma_i(x_{\mathcal{N}_i}) = \sum \alpha_i$

Time-Varying Distributed MPC Structure

Global problem: $\min \sum V^i(x_{\mathcal{N}_i}; \alpha_i)$

Local Problems:

$$
\begin{aligned}
V^i(x_{\mathcal{N}_i}; \alpha_i) = \min \quad & V_f^i(x(N)) + \sum_{k=0}^{N-1} l(x(k), u(k)) \\
\text{s.t.} \quad & x(0) = x_i \\
& x(k+1) = A_{ii}x(k) + B_i u(k) + \sum A_{ij} x_j(k) \\
& (x(k), u(k)) \in \mathcal{X} \times \mathcal{U} \\
& x(N) \in \mathcal{X}_f(\alpha_i)
\end{aligned}
$$

Distributed control (online for every subsystem):

1. Measure state
2. Solve global MPC problem by distributed optimisation, apply input u_i
3. Update $\alpha_i^+ = \alpha_i + x_{\mathcal{N}_i}^T(N)\Gamma_{\mathcal{N}_i} x_{\mathcal{N}_i}(N)$

Linear Quadratic case: Terminal cost synthesis

1. Local condition: $V_f^i(x_i^+) - V_f^i(x_i) \leq l_i(x_{\mathcal{N}_i}) + \gamma_i(x_{\mathcal{N}_i})$
2. Global condition: $\sum_{i=0}^{M} \gamma_i(x_{\mathcal{N}_i}) \leq 0$

Quadratic local cost functions: $l_i(x_{\mathcal{N}_i}, u_i) = x_{\mathcal{N}_i}^T Q_{\mathcal{N}_i} x_{\mathcal{N}_i} + u_i^T R_i u_i,\ Q_{\mathcal{N}_i}, R_i \succ 0$

Goal: Compute linear feedback law and relaxed quadratic Lyapunov functions

$V_i^f(x_i) = x_i^T P_i x_i,\ P_i \succ 0$	Quadratic relaxed Lyapunov function
$\gamma_i(x_{\mathcal{N}_i}) = x_{\mathcal{N}_i}^T \Gamma_i x_{\mathcal{N}_i}$	Indefinite coupling to neighbours
$u_i^f(x_{\mathcal{N}_i}) = K_{\mathcal{N}_i} x_{\mathcal{N}_i}$	Linear control law depends on neighbours

Linear Quadratic case: Terminal cost synthesis

1. Local condition: $V_f^i(x_i^+) - V_f^i(x_i) \le l_i(x_{\mathcal{N}_i}) + \gamma_i(x_{\mathcal{N}_i})$

2. Global condition: $\sum_{i=0}^{M} \gamma_i(x_{\mathcal{N}_i}) \le 0$

1. Local condition

$$(A_{\mathcal{N}_i} + B_i K_{\mathcal{N}_i})^T P_i (A_{\mathcal{N}_i} + B_i K_{\mathcal{N}_i}) - \bar{P}_i \preceq -Q_{\mathcal{N}_i} - K_{\mathcal{N}_i}^T R_i K_{\mathcal{N}_i} + \Gamma_{\mathcal{N}_i}$$

Nonlinear matrix inequality => LMI e.g., *[Zecevic, Siljak, 2010]*

2. Global condition:

$$\sum_{i=0}^{M} \gamma_i(x_{\mathcal{N}_i}) = \sum_{i=0}^{M} x^T \hat{\Gamma}_i x = 0 \qquad \Leftrightarrow \qquad \sum_{i=0}^{M} \hat{\Gamma}_i = 0$$

Sparse matrices with same structure as dynamic coupling

Offline synthesis: Solve one distributed convex LMI

Summary: Distributed MPC with Stability Guarantee

Distributed synthesis (offline):

1. Solve distributed SDP to compute:
   - Local relaxed Lyapunov functions P_i, $\Gamma_{\mathcal{N}_i}$
   - Local linear control laws $K_{\mathcal{N}_i}$
2. Solve distributed LP to compute initial feasible terminal size α

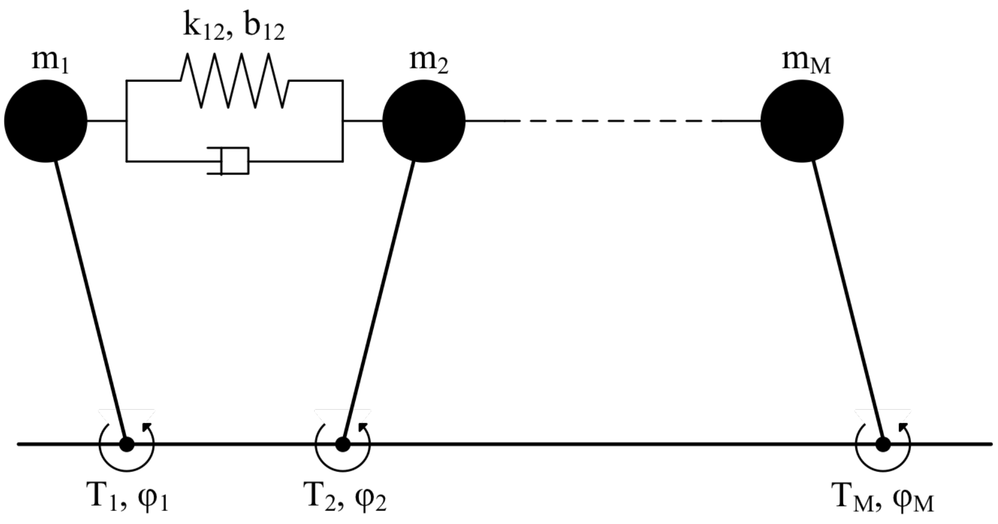
Distributed control (online for every subsystem):

1. Measure state
2. Solve global MPC problem by distributed optimisation, apply input u_i
3. Update $\alpha_i^+ = \alpha_i + x_{\mathcal{N}_i}^T(N)\Gamma_{\mathcal{N}_i}x_{\mathcal{N}_i}(N)$

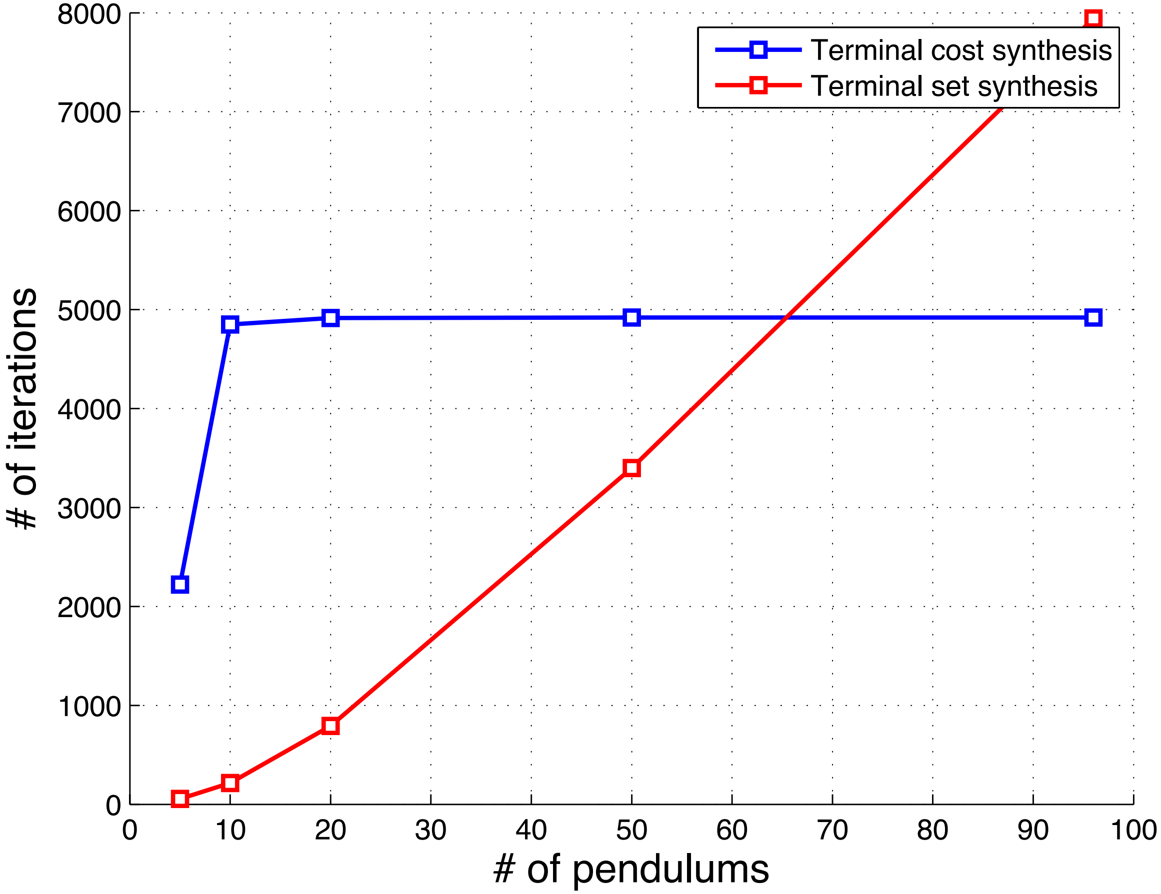
No central coordination required!

Computational example

- Chain of inverted pendulums (unstable)
- Linearized around the origin
- States: Angle and angular velocity of each pendulum
- Inputs: Torque at each pivot

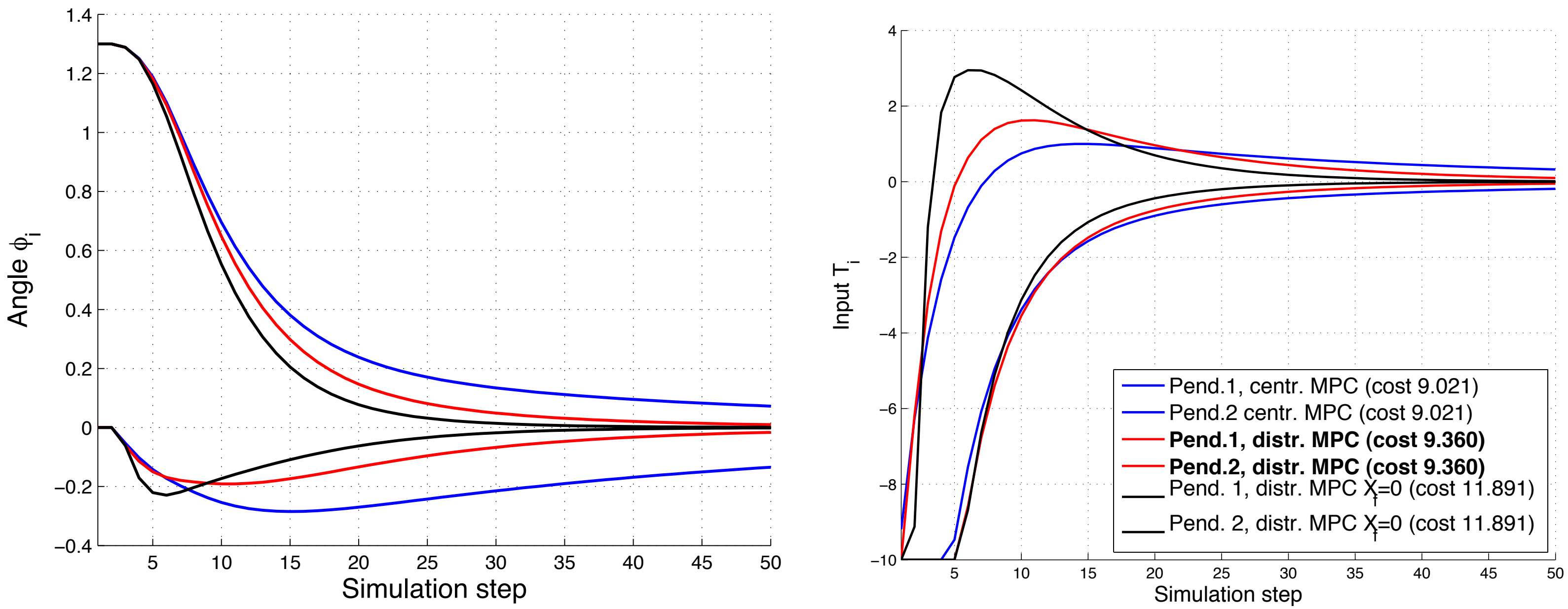


Synthesis: Scaling with chain length



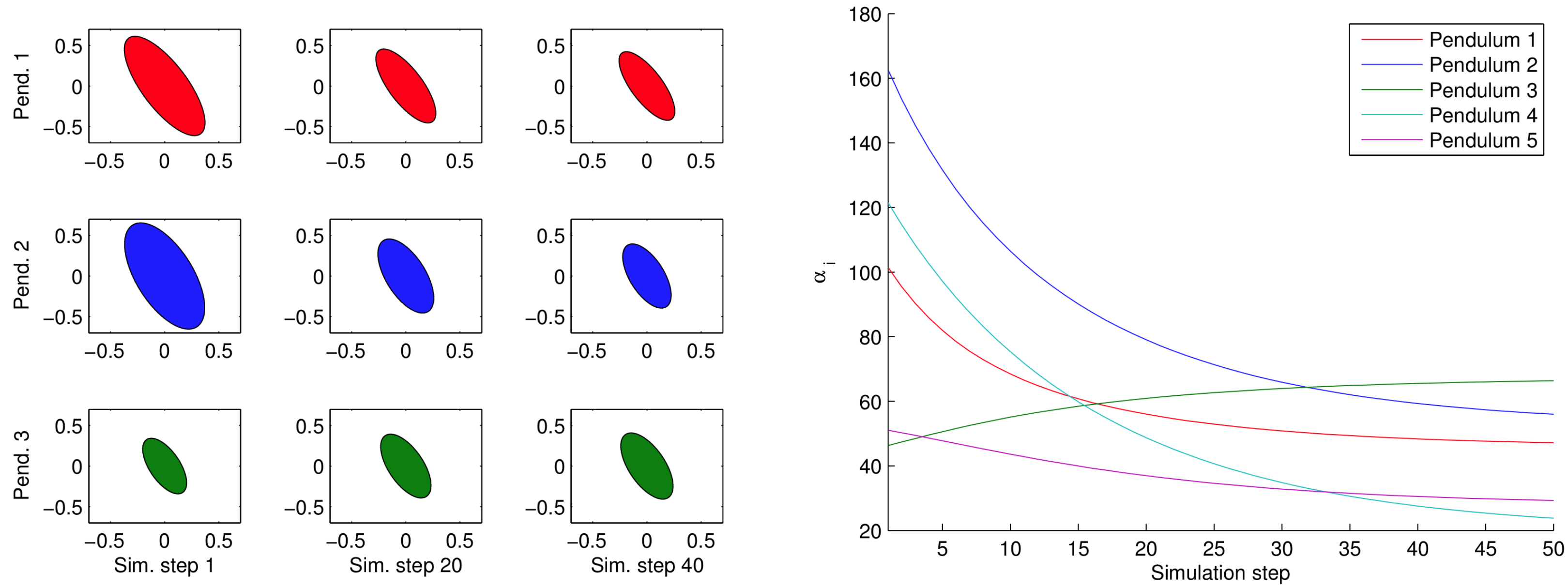
- Terminal cost (SDP):
  # Iterations saturates
- Terminal set (LP):
  Growth unbounded
- Possible explanation:
  LP has global constraint
  LMI coupled to neighbors

Closed-Loop Simulation



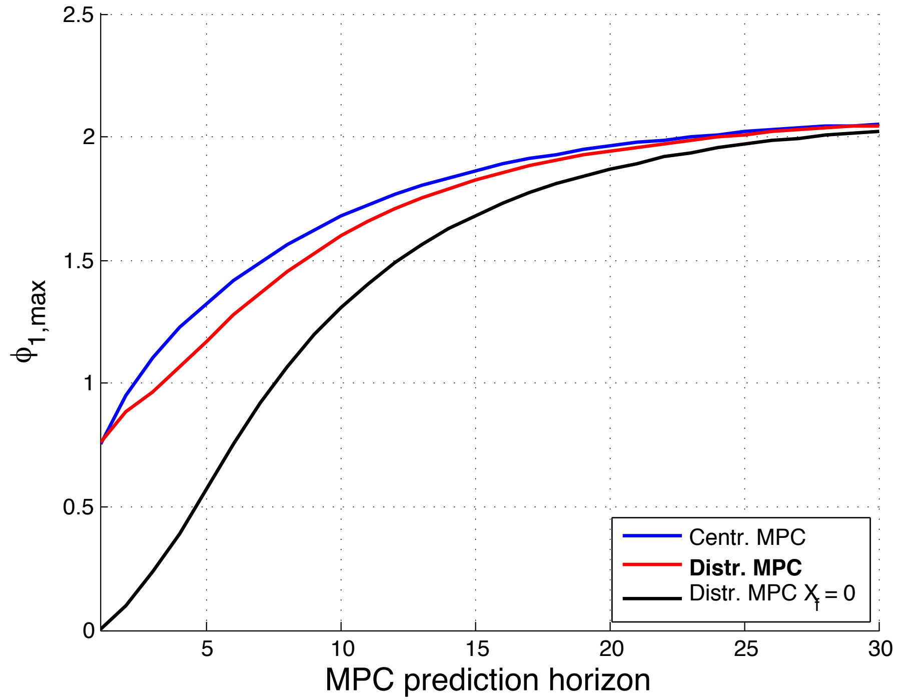
- 5 Pendulums, alternating direction method of multipliers, 100 iterations.
- Initially all pendulums in origin, only pendulum 1 is deflected.
- Cost of proposed method only **4%** higher than centralized MPC and **21%** lower than for a trivial terminal set.

Closed-loop simulation – Local Terminal Sets



Sizes of local terminal sets change dynamically

Region of Attraction



- Maximum feasible deflection of the first pendulum vs. prediction horizons
- *Short prediction horizons*: Region of attraction for proposed method significantly larger than for trivial terminal set
- *Long prediction horizons*: All methods converge to the same maximum control invariant set

Outline

Part I

Distributed dynamic invariant sets

☺ Larger region of attraction

Part II

Plug-and-play

☺ Adapt to changing networks

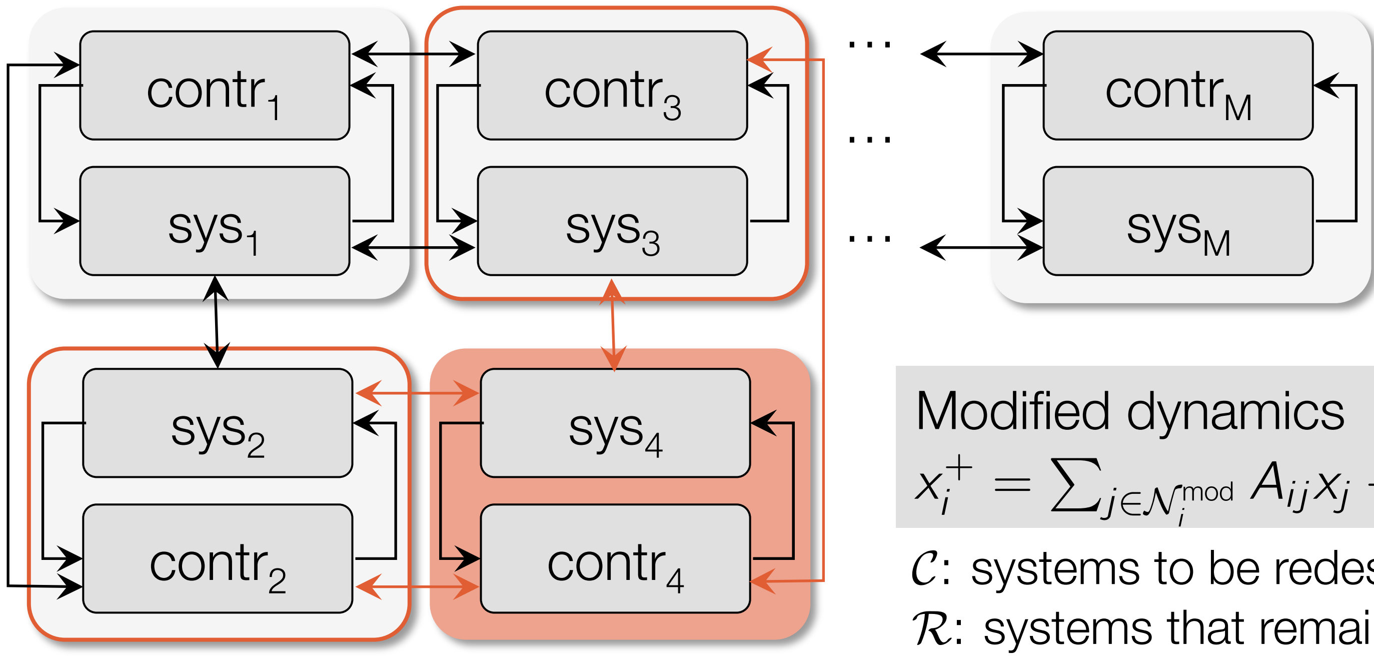
Part III

Accelerated distributed optimization

☺ (Towards) Real-time MPC

Plug and Play Predictive Control

Goal: Allow subsystems to join or leave the network



Modified dynamics

$x_i^+ = \sum_{j \in \mathcal{N}_i^{\text{mod}}} A_{ij} x_j + B_i u_i$

$\mathcal{C}$: systems to be redesigned

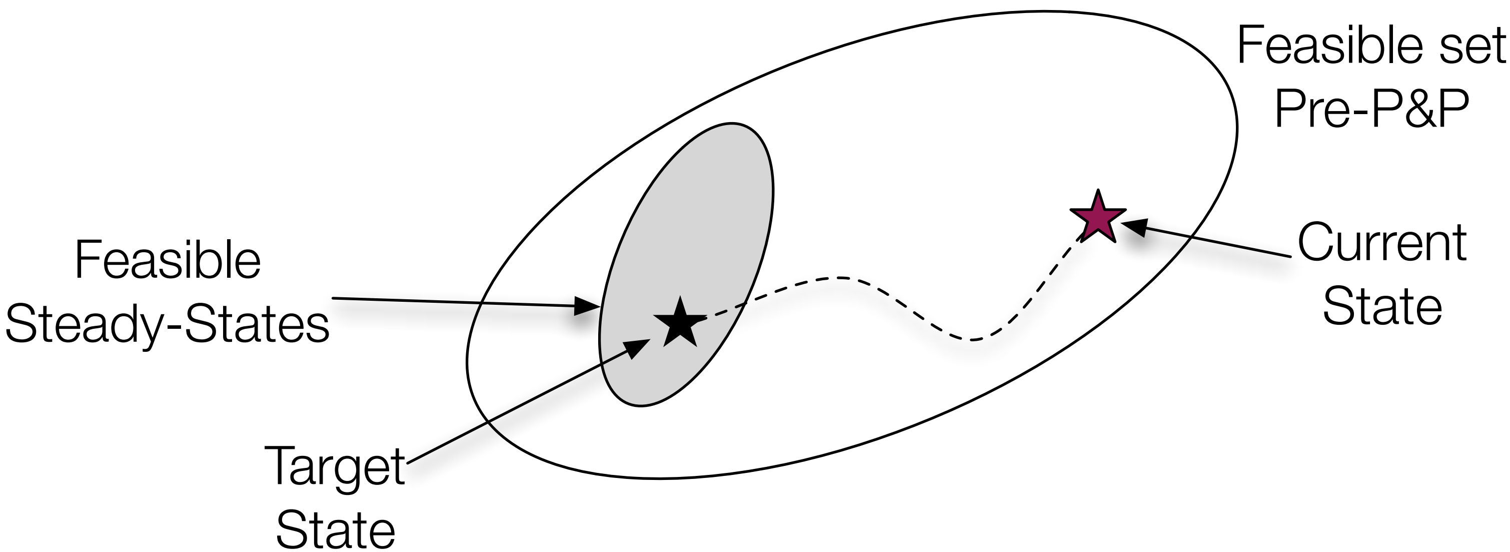
$\mathcal{R}$: systems that remain unchanged

Maintain stability and recursive feasibility during network changes:

- Adapt local control laws of subsystems and neighbours
- Ensure feasibility of the modified control laws

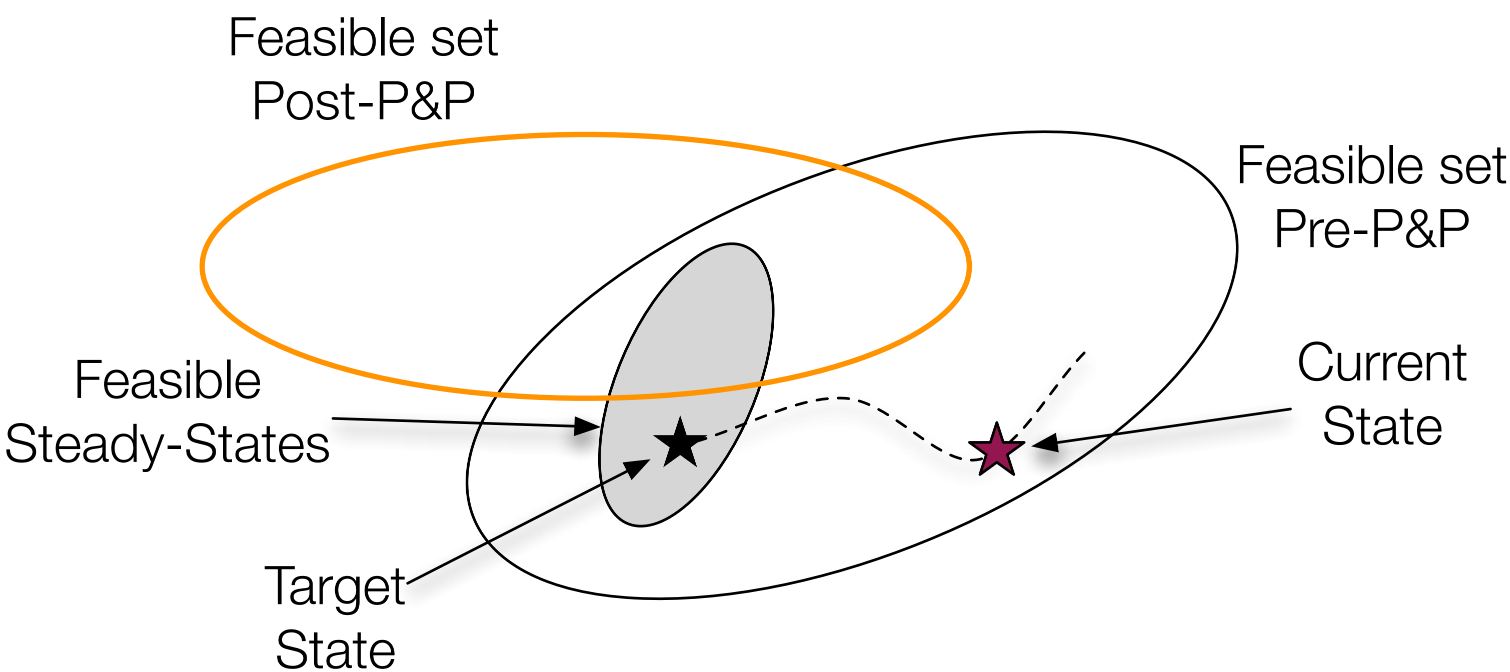
Plug and Play Outline

P&P Request	Redesign	Transition	Plug-in
• Notify neighbours • Only neighbours modify controller	• Compute terminal controllers / costs • Small SDP in background	• Track steady-state feasible for new & old system	• Use new (connected) control law



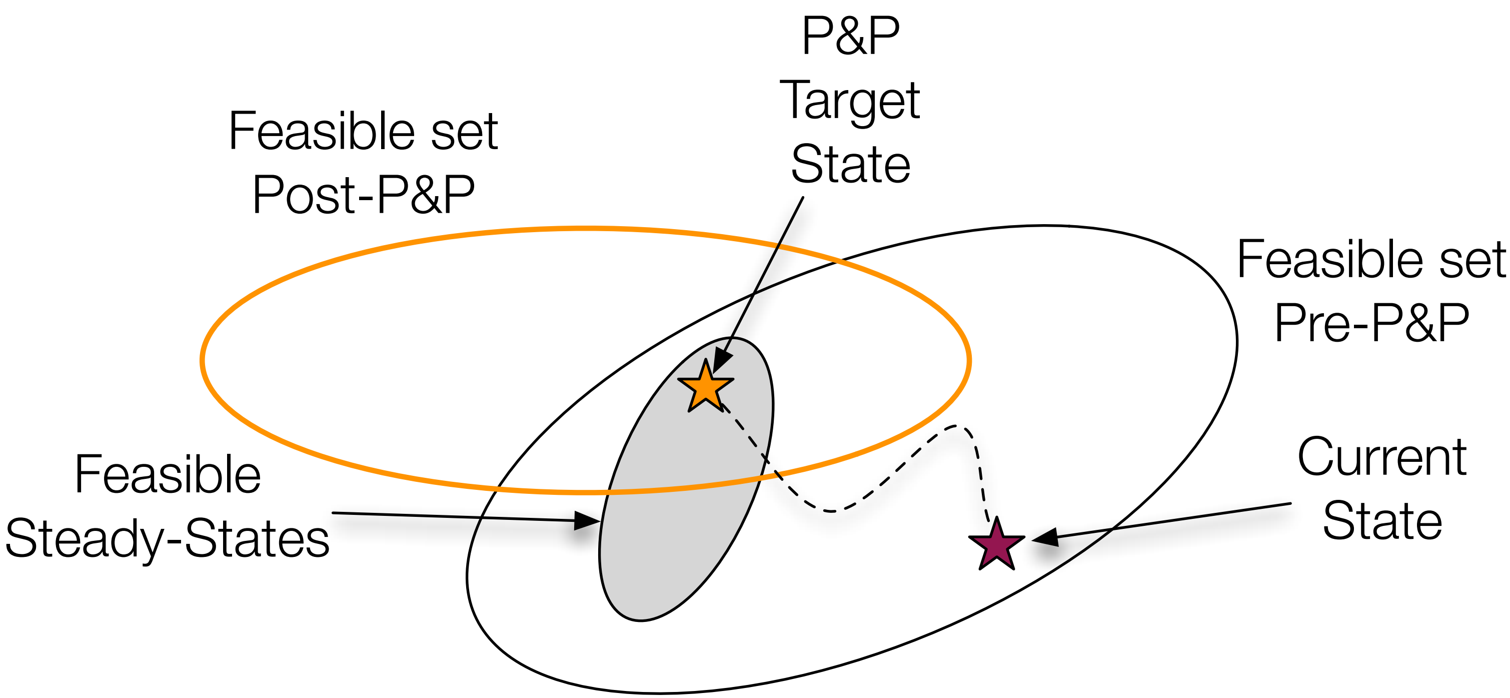
Plug and Play Outline

P&P Request	Redesign	Transition	Plug-in
• Notify neighbours • Only neighbours modify controller	• Compute terminal controllers / costs • Small SDP in background	• Track steady-state feasible for new & old system	• Use new (connected) control law



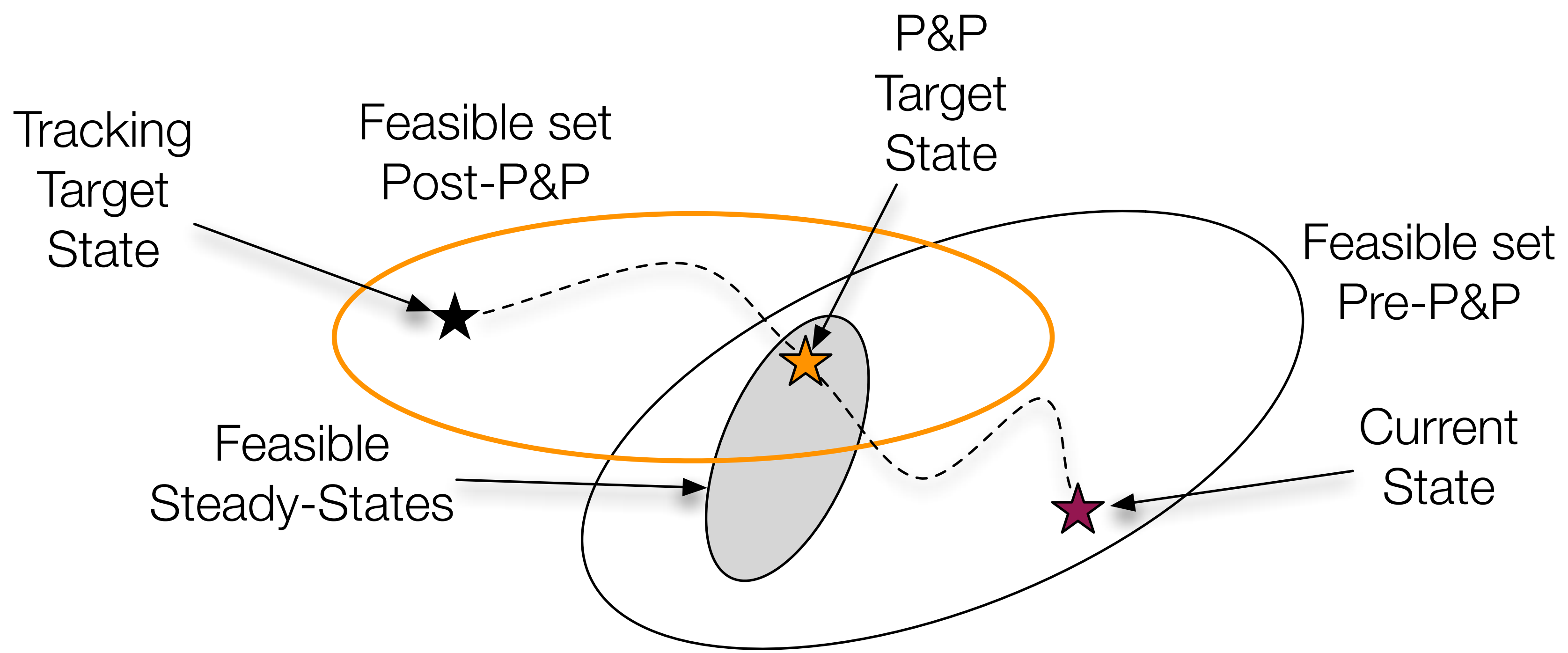
Plug and Play Outline

P&P Request	Redesign	Transition	Plug-in
• Notify neighbours • Only neighbours modify controller	• Compute terminal controllers / costs • Small SDP in background	• Track steady-state feasible for new & old system	• Use new (connected) control law



Plug and Play Outline

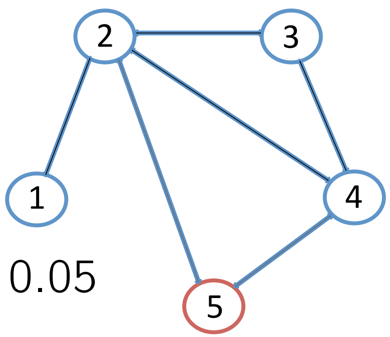
P&P Request	Redesign	Transition	Plug-in
• Notify neighbours • Only neighbours modify controller	• Compute terminal controllers / costs • Small SDP in background	• Track steady-state feasible for new & old system	• Use new (connected) control law



Computational example – Area Generation Control

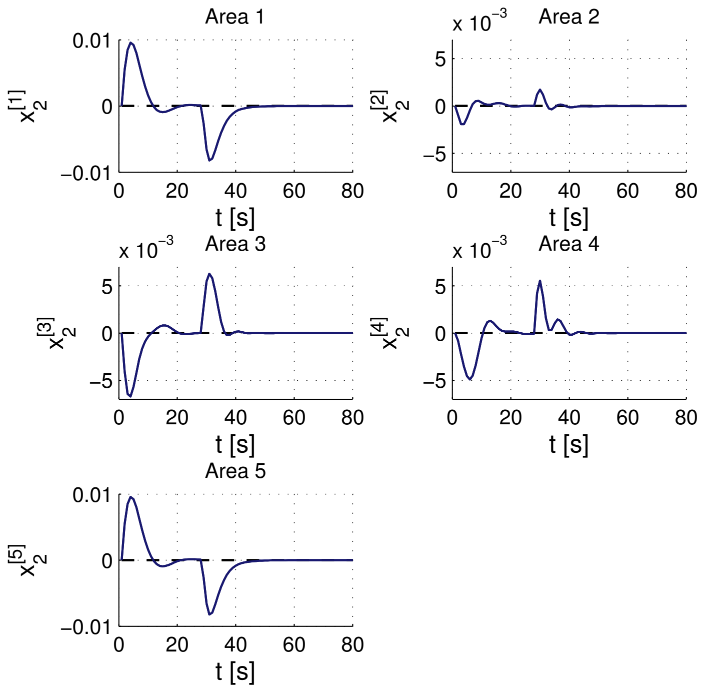
- Four power generation areas with load frequency control
- Model linearized around equilibrium *(Saadat, 2002; Riverso, et al. 2012)*

$$\dot{z}_i = \sum_{j \in \mathcal{N}_i} A_{ij} z_i + B_i v_i + L_i \Delta P_{L_i}$$

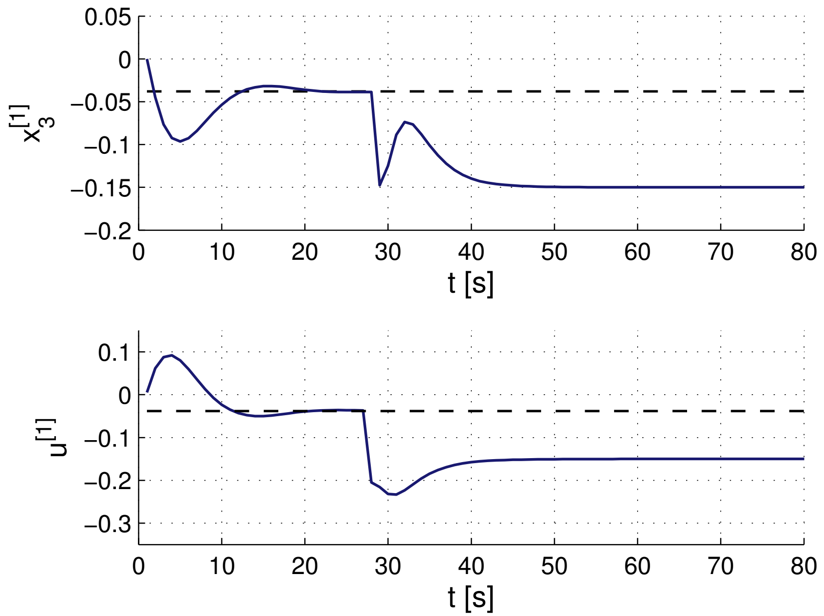


Goals: - Restore frequency, follow load change $\Delta P_{L_1} = -0.15, \Delta P_{L_3} = 0.05$

- Allow fifth area to join the network



Frequency deviation is controlled to zero



System is first regulated to steady-state and then to the origin

Outline

Part I

Distributed dynamic invariant sets

☺ Larger region of attraction

Part II

Plug-and-play

☺ Adapt to changing networks

Part III

Accelerated distributed optimization

☺ (Towards) Real-time MPC

Distributed Optimization ⇒ First-Order Method

$$\begin{aligned} \min \ & \sum f_i(y_i) \\ \text{s.t. } & y_i \in Y_i \\ & \sum A_i y_i = c \end{aligned}$$

Distributed optimization requires that the problem is structured

Example: Dual Decomposition

$$g(\lambda) = \min_{y_i \in Y_i} \sum f_i(y_i) + \lambda^T \left(\sum A_i y_i - c \right) = \sum \min_{y_i \in Y_i} f_i(y_i) + \lambda^T A_i y_i$$

Gradient of the dual function: $\nabla g(\lambda) = \sum A_i y_i^\star(\lambda) - c$

Gradient-based approach

$\lambda^+ = \lambda + \alpha \nabla g(\lambda)$

$\left.\begin{array}{l} \text{Optimal values } y_i^* \rightarrow \text{Local optimization} \\ \text{Dual update} \rightarrow \text{Consensus} \end{array}\right.$

Many variants on this theme (ADMM, AMA, FISTA, ...)

Distributing an MPC Problem

Centralized MPC problem for distributed system:

$$J(x) = \min \sum_k J^k(x_0^k, \bar{x}^k)$$
$$\text{s.t. } \bar{x}^k = x$$

Local copy of neighbour's state

Current state

Proximal form, used in most accelerated variants:

$$J^k(x_0^k, \bar{x}^k) = \min_{x,u} \sum l(x_i, u_i) + \rho \sum_j \|x_i^j - \bar{x}_i^j\|_2^2$$
$$\text{s.t. } x_{i+1} = Ax_i + Bu_i + \sum A^j x_i^j$$
$$x_i \in \mathcal{X}, \ \ u_i \in \mathcal{U}$$
$$x_0 = x^i$$

"Track" impact of neighbours on own trajectory

Optimize over own trajectory and neighbour's impact

Local problems are (almost) standard MPC for tracking

Distributed Optimization

Total time to compute control law =
(Number of global iterations) * (Time to solve one local problem)

Global optimization problem
- First-order information
- Iterations cheap (requires comm)
- Variable number of iterations

Minimize number of iterations
- Pre-conditioning
- Formulation (linear convergence)
- Warm starting

Local optimization problems
- Second-order information
- Iterations expensive
- Constant number of iterations

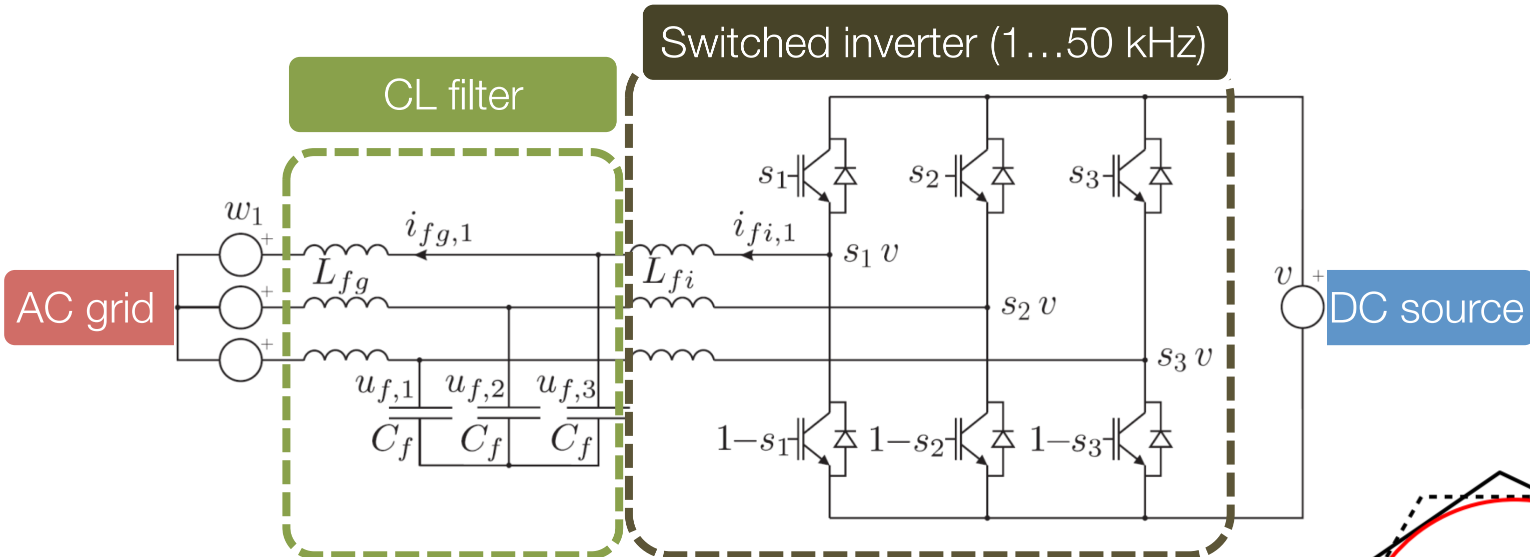
Accelerate single iteration
- Fast linear algebra
- Exploit structure of MPC problems
- Code-generation

SPLIT
G. Stathopolous, Y. Pu & A. Szucs

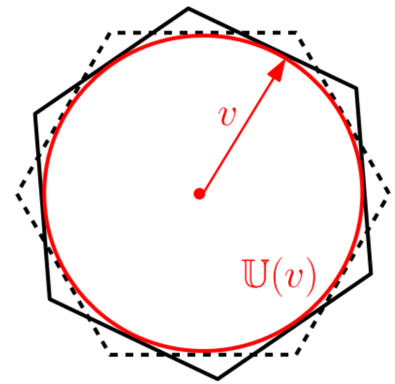
FiOrdOs
F. Ullmann & S. Richter

FORCES
A. Domahidi & M. Zeilinger

Example: AC/DC Converter

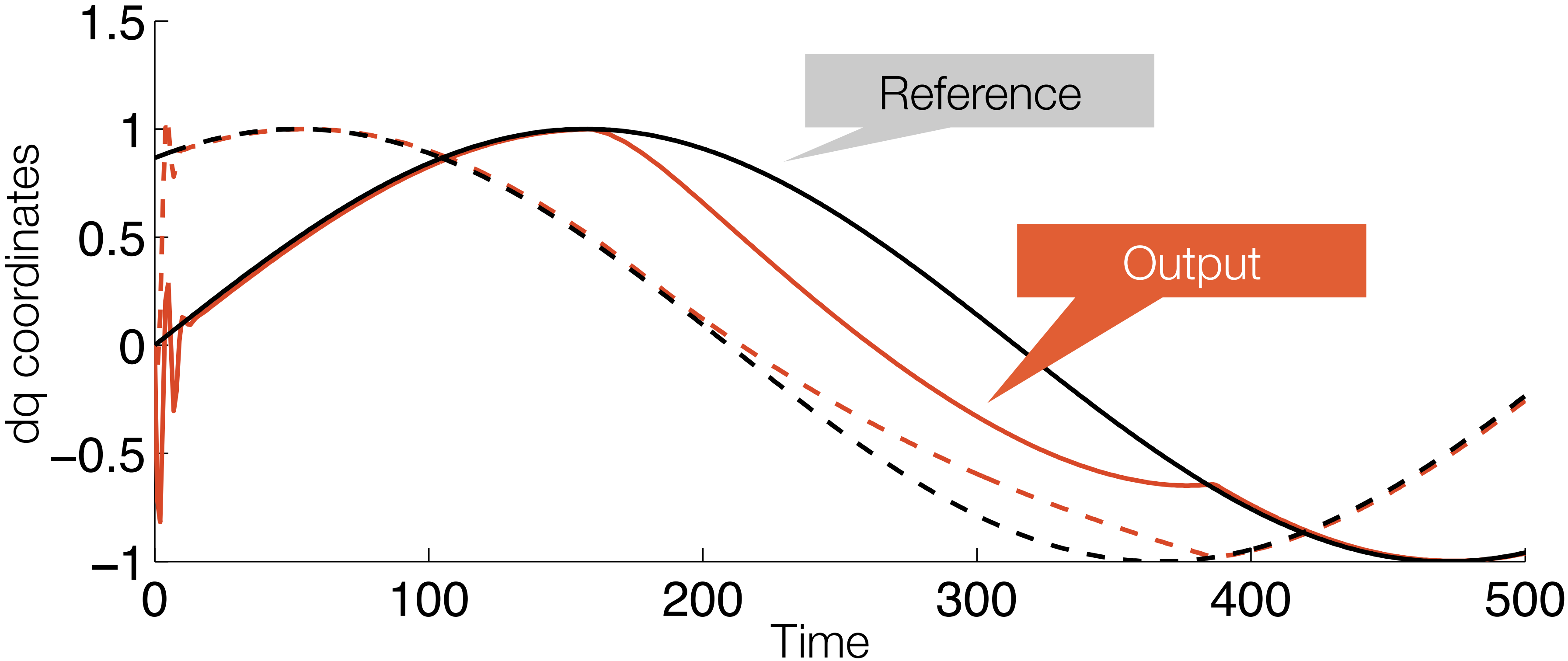


$$\min \frac{1}{2} u^T H u + g(x, x_{ss}, u_{ss}, w)^T u$$
$$\text{s.t.}\, u_k \in \mathbb{U}(\nu) - u_{ss}\ ,\ k = 0, \ldots, N-1$$



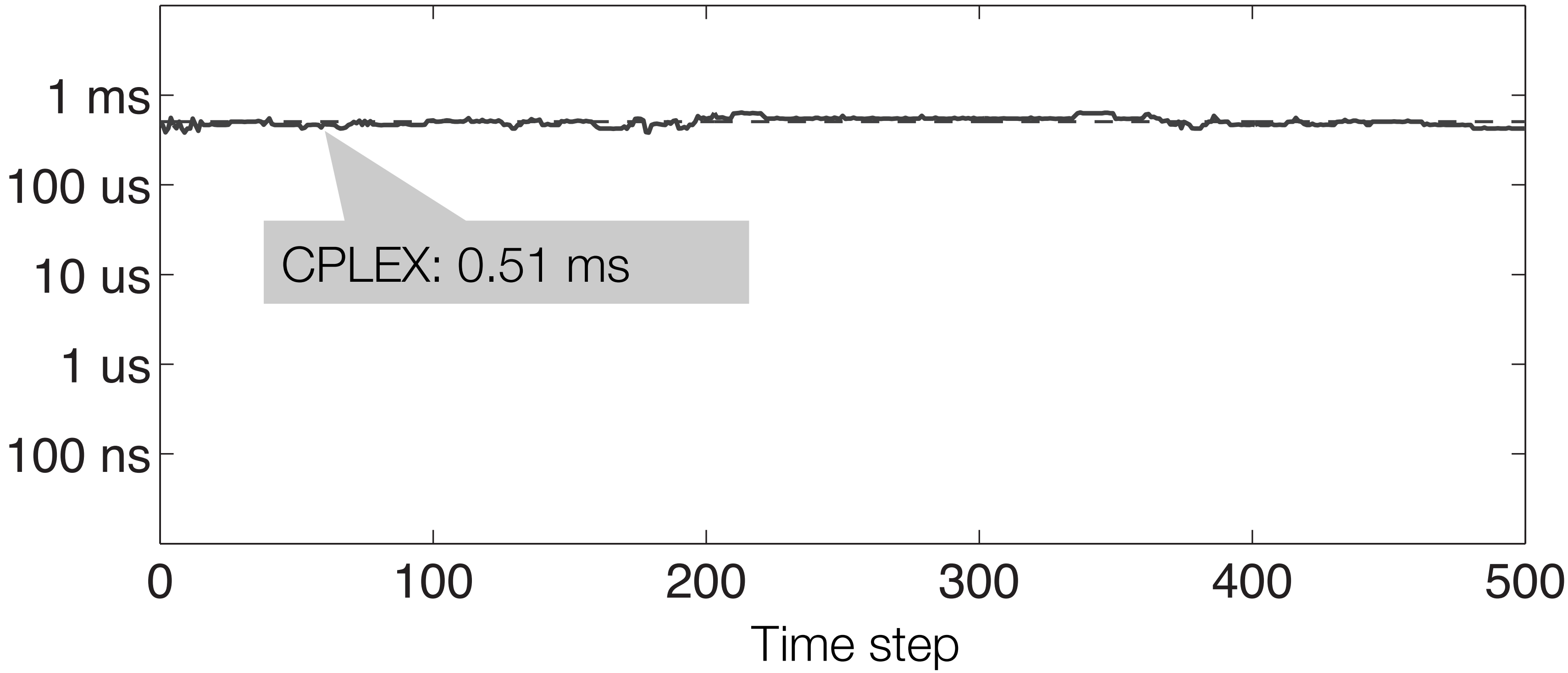

```
U   = EssBall(2, 'r','param', 'shift','param', 'r_max', 1);
U_N = SimpleSet(N);
U_N.addSet(1:N, U);
Prob = OptProb('H',H, 'g','param', 'X',U_N);
```

AC/DC Converter: Simulation Results

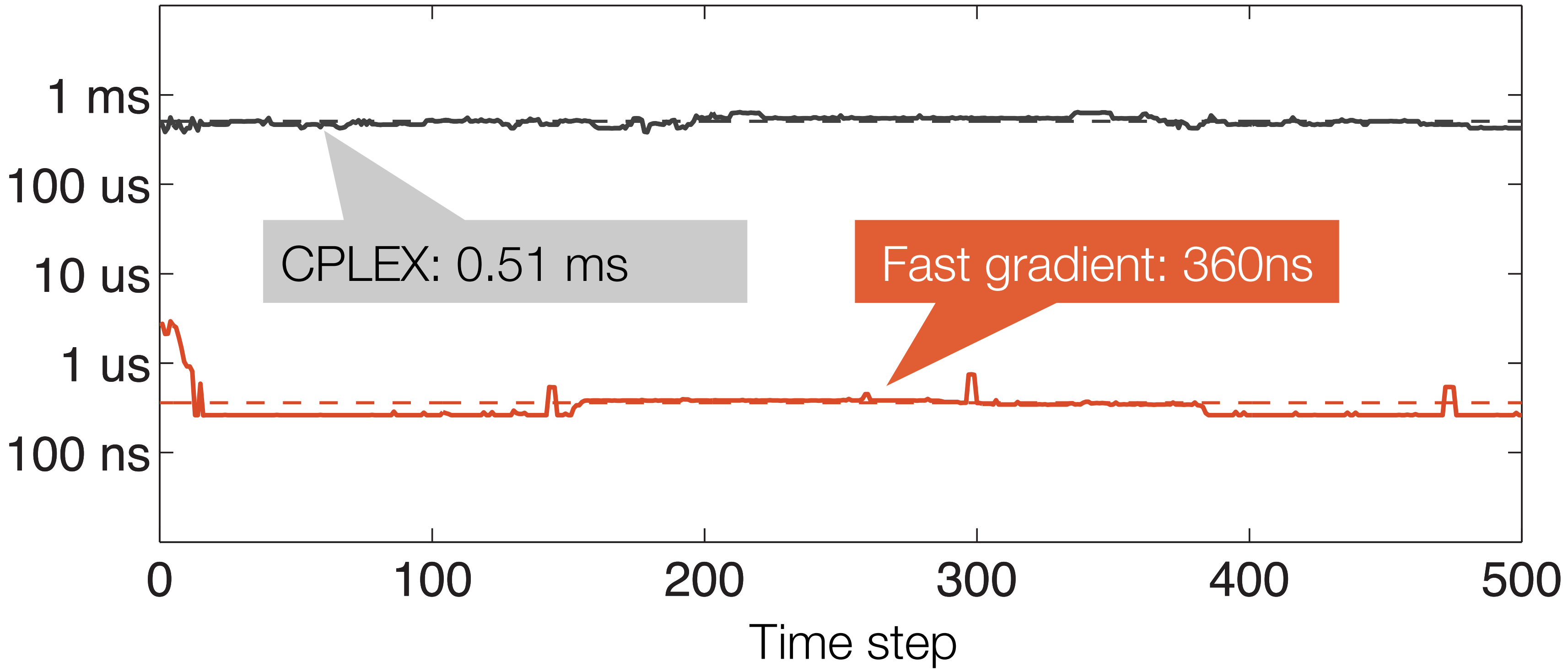


Specified accuracy of 10^{-3} produces good tracking results

Performance of Auto-Tuned FGM on 2.5GHz PC



Performance of Auto-Tuned FGM on 2.5GHz PC



On average 1400x faster than CPLEX

Distributed Optimization

Total time to compute control law =
(Number of global iterations) * (Time to solve one local problem)

Global optimization problem

- First-order information
- Iterations cheap (requires comm)
- Variable number of iterations

Minimize number of iterations

- Pre-conditioning
- Formulation (linear convergence)
- Warm starting

Local optimization problems

- Second-order information
- Iterations expensive
- Constant number of iterations

Accelerate single iteration

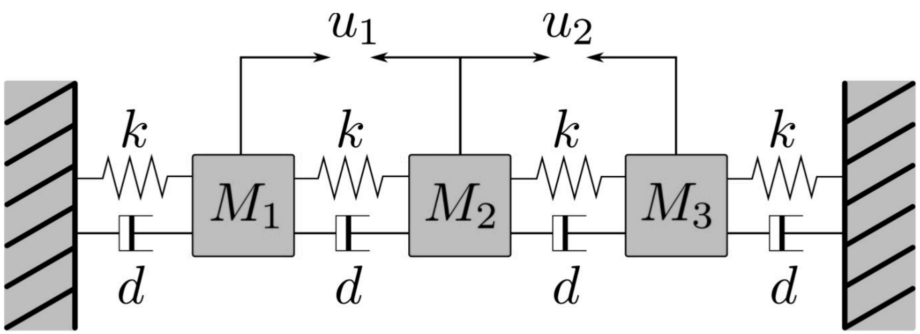
- Fast linear algebra
- Exploit structure of MPC problems
- Code-generation

SPLIT — *G. Stathopolous, Y. Pu & A. Szucs*

FiOrdOs — *F. Ullmann & S. Richter*

FORCES — *A. Domahidi & M. Zeilinger*

Benchmark problems



Problem QP:
QP with box constr./diagonal cost
(no stability guarantees)

$$\min_{\mathbf{u}} V_N(x, \mathbf{u}) := \sum_{i=0}^{N-1} x_i^T Q x_i + u_i^T R u_i + V_f(x_N)$$

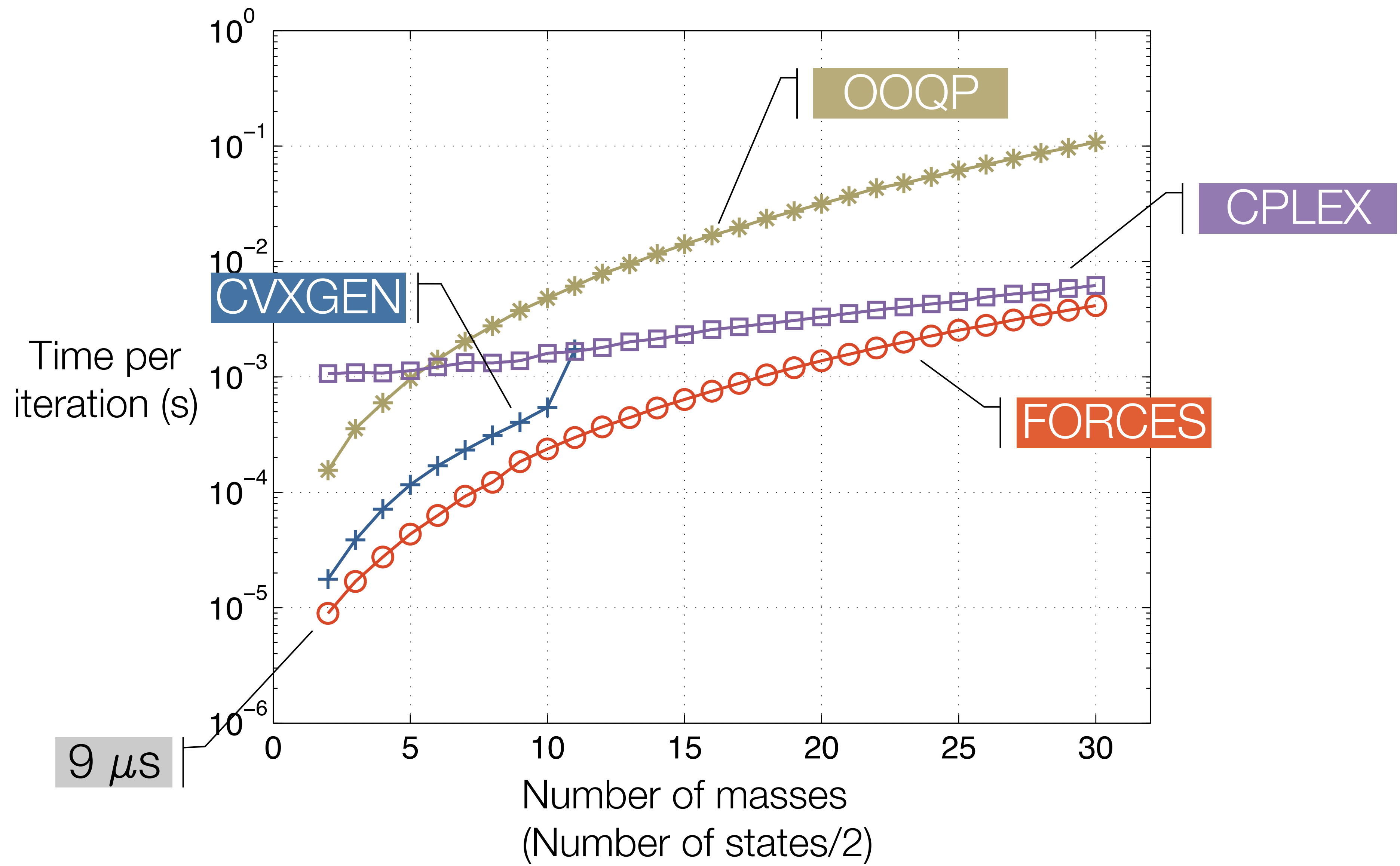
$$\text{s.t. } x_0 = x(0), \qquad V_f(x_N) = x_N^T Q x_N$$

$$x_{i+1} = A_i x_i + B_i u_i$$

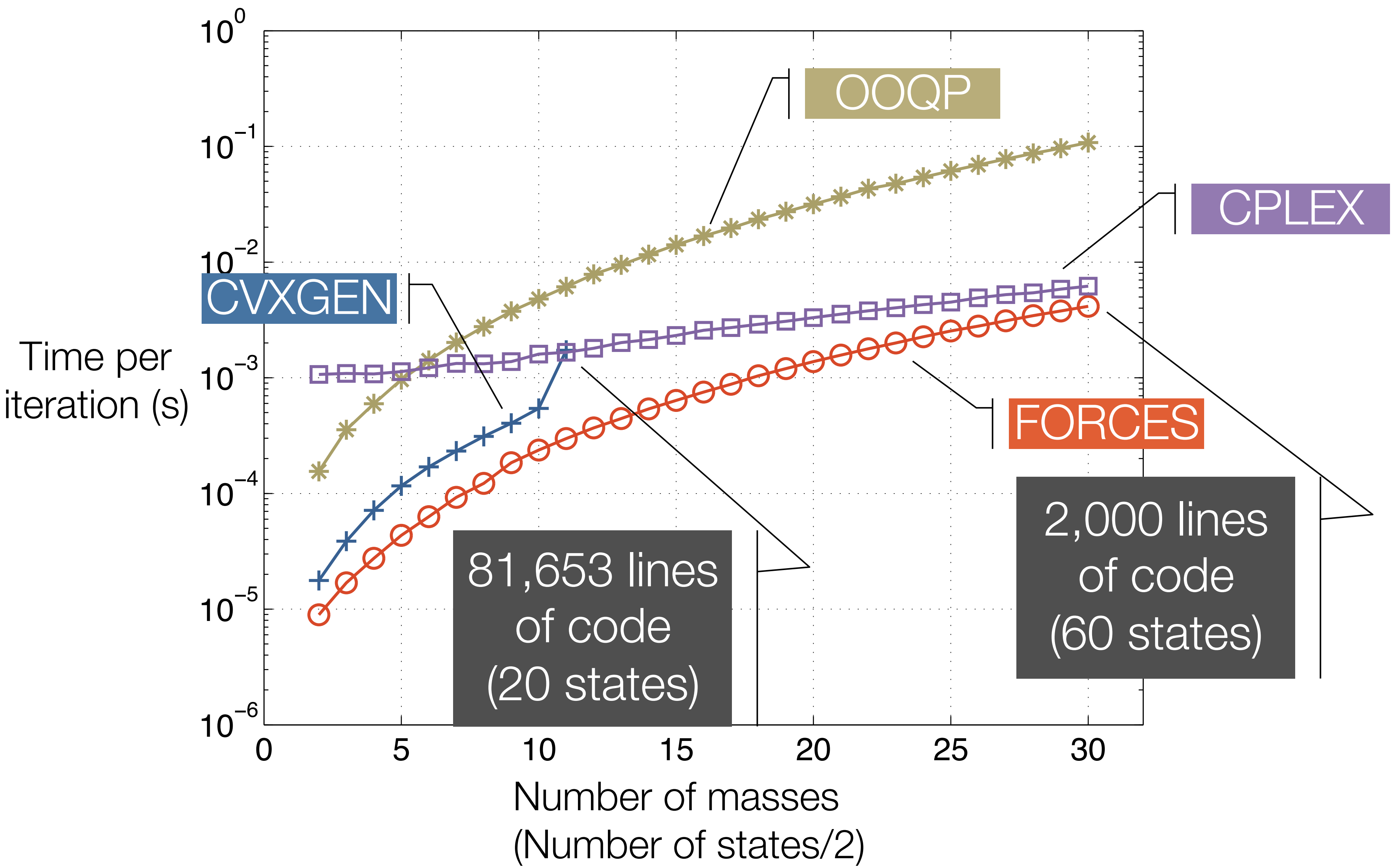
$$-4 \cdot \mathbf{1} \leq x_i \leq 4 \cdot \mathbf{1}$$

$$-0.5 \cdot \mathbf{1} \leq u_i \leq 0.5 \cdot \mathbf{1}$$

Computation times on PC for QP



Computation times on PC for QP

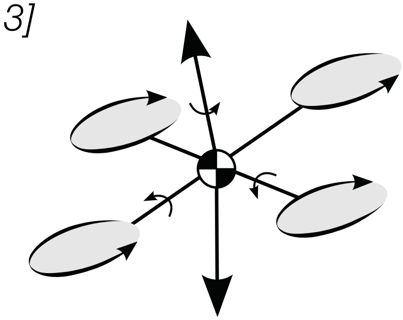


Some Early Users of FORCES

Agressive Quadrocopter Maneuvers

[M. Muller & R. D'Andrea, 2013]

IDSC



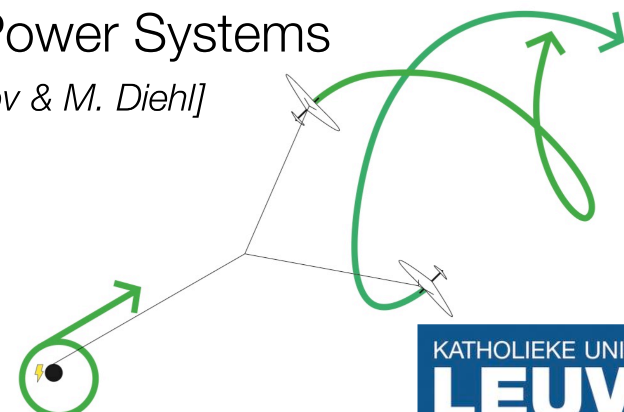
Wind Turbine Control

[Marc Guadayol, 2013]

ALSTOM

Kite Power Systems

[M. Vukov & M. Diehl]



KATHOLIEKE UNIVERSITEIT
LEUVEN

Belt-Drive Control

[Kim Listmann, 2013]

ABB

Berkeley
UNIVERSITY OF CALIFORNIA

DTU

Danfoss

TUM
TECHNISCHE
UNIVERSITÄT
MÜNCHEN

UNIVERSITY OF
CAMBRIDGE

STANFORD
UNIVERSITY

LMS®
ENGINEERING INNOVATION

Summary – Distributed MPC

Main limitation of MPC theory in a distributed setting:

- Invariant sets and Lyapunov functions couple all systems

Key idea:

- Structured Lyapunov functions and dynamic invariant sets guarantee stability and invariance by design, without introducing additional coupling

Extensions / References

Synthesis and control via distributed optimization	*[Conte, et al., ACC 2012], [Conte et al, CDC 2012]*
Robust Tube-Based MPC	*[Conte, et al., ECC 2013]*
Tracking MPC	*[Conte, et al. CDC 2013]*
Plug and play MPC	*[Zeilinger, et al., CDC 2013]*